# Проектирование интерфейса как часть разработки ТЗ

Владислав Головач, Александр Белышкин

Внедрение систем автоматизации бизнеса, как знает любой вовлеченный в эту область специалист, отнюдь не является легким делом. Если само создание системы, вообще говоря, технически не очень сложно (к примеру, нельзя сказать, что среднестатистическая система наполнена сложнейшими алгоритмами), то внедрение требует от автоматизатора недюжинной квалификации, редкого упорства и изворотливости. При этом корни многих проблем находятся в техническом задании. Как говорится, «что задумали, то и сделали», но потом иногда оказывается, что задумали-то и неправильно. Для решения проблем, возникающих при создании ТЗ, а проявляющихся при внедрении, придумано множество технологий и методов однако само их количество свидетельствует о том, что ни один метод к полному успеху не приводит. Кроме того, многие методы имеют принципиальный недостаток – они увеличивают объем части работы, пусть и ради экономии на другом этапе, и требуют серьезных инвестиций в обучение сотрудников (характерный пример – RUP). Существует, однако, подход, который не требует особой квалификации сотрудников, значительно облегчает внедрение, не увеличивая объем работ по разработке ТЗ.

Сущность подхода может быть описана одной фразой — проектирование интерфейса не есть часть процесса разработки, а часть процесса создания спецификаций на систему.

Здесь необходимо сделать два важных уточнения. Во-первых, интерфейс всё равно будет разработан (практика показывает, что заказчики почему-то неохотно оплачивают функциональность без интерфейса). Во-вторых, от проектирования интерфейса ничего особого не требуется — на него могут быть выделены те же ресурсы, что и в случае обычной разработки, равно как и получиться он может таким же. Авторам, зарабатывающим себе на жизнь разработкой эргономичных интерфейсов, неприятно это говорить, но интерфейс может быть даже не эргономичным, все равно внедрение будет облегчено; разумеется, в случае эргономичного интерфейса внедрение будет ещё более простым, но такой интерфейс дороже и дольше делается.

Предлагаемый подход позволяет решить следующие проблемы:

Устранить различия во взглядах на постановку задачи заказчика и исполнителя. Спецификации на сколько-нибудь сложные системы слишком абстрактны. Их с трудом удерживают в голове даже авторы, а до конца не понимает никто, в особенности ключевое лицо — заказчик. Для него эта спецификация ничем не отличается от сакральных письмен (многие даже предполагают, что непонятные ТЗ предназначены для того, чтобы произвести на них впечатление и содрать побольше денег). Нет разницы, что подсунут ему разработчики, реальное ТЗ или инструкцию к газонокосилке на фарси — все одинаково непонятно. Наивно предполагать, что заказчик, легко подписавший такое ТЗ, также легко примет разработанную систему. Прототипы интерфейса являются тем единственным документом, который заказчик может понять и оценить. А поняв и оценив — сознательно подписать.

Облегчить процесс внедрения системы. Весомая часть проблем внедрения в качественно выполненном проекте приходится на интерфейс, созданный формально правильно, но неадекватно представлениям заказчика. Не существует вида ТЗ, кроме собственно прототипа интерфейса, который бы мог интегрировать такого рода требования. Наглядный пример: в любом ТЗ можно прописать, что «в системе есть адресная книга, которая состоит из таких-то данных и таких-то функций». Но невозможно формализовать уже в ТЗ, как эта адресная книга должна реально работать (какие-то функции нужно «вытащить» наверх, какие-то можно «задвинуть»), как, в конце концов, эта адресная книга должна выглядеть. При этом апелляции исполнителя к подписанному техническому заданию – дескать, вот же перечисленные функции... вот они все налицо — как правило, не срабатывают, поскольку при известной изворотливости в контексте пользовательского интерфейса проинтерпретировать ТЗ всегда можно очень по-разному. Только заранее спроектированный интерфейс позволяет застраховаться от такого рода претензий.

Сократить число доработок системы, вызванных несоответствием ее функциональности ожиданиям клиента. Только увидев саму систему, заказчик может реально понять ее возможности, равно как и оценить собственные потребности. Для заказчика программный продукт и его интерфейс совершенно тождественны. Следовательно, показав заказчику интерфейс на стадии подготовки ТЗ, можно снизить количество и объем переделок, потребность в которых возникает из-за расхождений ожиданий заказчика с запланированной в ТЗ функциональностью системы. (Нужно, впрочем, отметить, что такие переделки чаще всего не проблематичны для разработчиков, которые обычно настаивают на дополнительной оплате этих услуг.)

Снять риск необходимости доработки функциональности системы, из-за неудовлетворенности заказчика предложенным интерфейсом. При разработке интерфейса нет решительно никакой гарантии того, что он будет принят заказчиком. Описание функций системы бинарно, функция может быть, может не быть. Доказательство её наличия редко требует аргументации. Интерфейс же может быть либо достаточно хорошим, либо недостаточно хорошим. Когда в дело вступают относительные термины, все усложняется, что может приводить в возникновению конфликтных ситуаций. Нечего и говорить, что при переделке недостаточно хорошего интерфейса функциональность системы, которая уже есть, меняется тоже, причем без оплаты труда разработчика.

Таким образом, есть объективная польза в том, чтобы рассматривать проектирование интерфейса не как стадию разработки, а как стадию создания ТЗ. Но как это сделать? На первый взгляд, задача кажется трудноразрешимой, частично с организационной, частично с технической сторон.

Сначала об организационной стороне. На первый взгляд заказчика трудно убедить отказаться от мысли, что делать что-то «реальное» надо сразу после подписания договора. Однако практика показывает, что промежуточные наглядные результаты работы над системой, а именно прототипы интерфейса, продемонстрированные уже на второй день работы, а не через несколько недель, приводят клиента в благодушество. В отличие от обычного ТЗ, работа над которым заказчику реально не видна («ну что там, пара пунктов добавилась») прототипы интерфейса легко понятны и прогресс в работе явно заметен. Вторая организационная проблема связана с необходимостью подписывать два договора: на создание ТЗ (читай — интерфейса) и на разработку функциональности системы. Причем подписание второго договора откладывается на определенный срок, необходимый для разработки интерфейса, что растягивает проект во времени. В принципе, эта проблема неразрешима, но, с другой стороны, здесь многое зависит от её восприятия: да, договора два, но зато второй договор получается значительно более точным (уже имея интерфейс, легче оценить трудозатраты).

Техническая проблема связана с трудностями прототипирования. В обычном режиме работы интерфейс создается уже в средстве разработки, создавать же прототип таким образом нерентабельно. Интерфейс создается через множество итераций, а переделывать уже сделанное уже дорого. Сравнительно недавно появились специальные средства для прототипирования интерфейса (например, Norpath Studio), но пока они довольно сырые. Выход — использование неспециализированных графических редакторов. Ещё несколько лет назад основным таким редактором являлся MS PowerPoint, сейчас же наиболее удобным следует признать MS Visio. Сложные экранные формы, впрочем, до сих пор удобнее просто рисовать на бумаге.

И, наконец, главная проблема. Удлинение процесса разработки ТЗ часто воспринимается самими разработчиками как безусловное зло — привычка сначала делать, а уж потом думать, традиционно сильна в российском IT-бизнесе. Увы, изменить этот обычай может только «опыт, сын ошибок трудных». Пока, во всяком случае…

Конечно, проектирование интерфеса на этапе разработки спецификаций системы не является панацеей. Такой подход не позволяет улучшить качество разработки в принципе, например, он вовсе не уменьшает количество ошибок программирования . Более того, он не всегда применим. Интерфейс сложной системы невозможно с самого начала спроектировать полностью: придется сначала делать работающую бета-версию и окончательно править интерфейс уже на её основе. Кроме того, во многих проектах из-за не зависящих ни от кого причин не получается растягивать процесс создания ТЗ (заказчик хочет увидеть какие-нибудь результаты уже завтра). Однако, учитывая низкие «входные» требования к применению предложенного метода (несравнимые, например, с волокитой и бюрократией, обусловленной использованием UML), проектирование интерфейсов на стадии подготовки спецификаций почти всегда является крайне успешным методом решения проблем внедрения.